AMENDMENTS TO THE SPECIFICATION

Please amend ¶ [0001] of the Specification as follows:

[0001] A virtual machine monitor ("VMM") creates an environment that allows multiple operating systems to run simultaneously on the same computer hardware. In such an environment, applications written for different operating systems (e.g., ~~Windows, Linux~~ WINDOWS® operating system, LINUX® operating system) can be run simultaneously on the same hardware.

Please amend ¶ [0019] of the Specification as follows:

[0019] Reference is made to FIG. 1 [[1a]], which illustrates hardware and software layers 110 and 111 of a computer 100. The hardware layer 110 includes a central processing unit (CPU), memory and I/O devices. Exemplary I/O devices include, without limitation, network adapters, SCSI controllers, video cards, host bus adapters, and serial port adapters. The memory refers to the memory that is internal to the computer (e.g., internal memory cache, main system memory) as opposed to external storage devices such as disk drives. The software layer 111 includes an operating system or OS instances 112, applications 114, and a virtual machine monitor 116. During execution, the software for the software layer 111 can be stored in "articles" such as the memory; and during distribution, the software can be stored in articles such as external devices, removable storage media (e.g., optical discs), etc.

Please amend ¶ [0038] of the Specification as follows:

[0038] If an OS boots on the hardware, and claims all the memory in the hardware (see, e.g., FIG. 2b), the VMM performs steps to gain control over that memory. As a first example of gaining control over the memory, the VMM may perform runtime virtualization of the physical memory used by the OS. Such runtime virtualization is disclosed in assignee's U.S. Patent Application Serial No. 10/677,159 [[_____]] filed October 1, 2003, U.S. Patent Publication No. 2005/0076156, [[_____]] ~~(attorney docket no. 200300561-1)~~ and incorporated herein by reference. Because the CPU is already virtualized, the VMM already has control over the hardware to perform the method disclosed therein.

Please amend ¶ [0040] of the Specification as follows:

[0040] The VMM can also perform steps to gain control of the I/O. As a first example, the VMM can virtualize I/O devices at runtime by commencing I/O emulation at runtime as described in U.S. Patent Application Serial No. 10/676,922 [[_____]] filed October 1, 2003, [[_____]]~~(attorney docket no. 200309154-1)~~ U.S. Patent Publication No. 2005/0076155, and incorporated herein by reference. Because the CPU is already virtualized, the VMM already has sufficient control over the hardware to perform the method disclosed therein.

Please amend ¶ [0044] of the Specification as follows:

[0044] Reference is made to FIG. 5, which illustrates how the dual-mode drivers may be used to interpose the VMM on an I/O device. First, interrupts are disabled (510). Next, the dual-mode driver for this device in the OS is instructed to switch to the virtual mode of operation (512). The driver can be so instructed by the kernel module in the OS, by the VMM already running on the CPU, or by an application that calls an IOCTL (input/output control) routine in the driver. Next, interrupts for the device are redirected to handlers in the VMM, if they were not already redirected when the CPU was virtualized (514). Finally, interrupts are re-enabled (516). From that point on, the OS's dual-mode driver performs I/O by calling the VMM's driver for the device

Please amend ¶¶ [0047] to [0048] of the Specification as follows:

[0047] If dual-mode drivers are not used, the VMM can devirtualize the device by ceasing emulation of the I/O device at runtime as disclosed in U.S. Patent Publication ~~Serial~~ No. 2005/0076155 [[_____]] ~~(attorney docket no. 200309154-1)~~.

[0048] The VMM may devirtualize memory as disclosed in U.S. Patent Publication ~~Serial~~ No. 2005/0076156 [[_____]] ~~(attorney docket no. 200300561-1)~~. The VMM should return control of devirtualized memory to the OS.

Please amend ¶ [0052] of the Specification as follows:

[0052] Reference is now made to FIG. 8. After the hardware has been fully devirtualized, the VMM's activities are completely stopped, control of the hardware is passed to the OS (810), and the VMM can be removed from the system. First, control of memory is returned to the OS (812). Next, control of I/O devices is returned to the OS (814). Next, the CPU is devirtualized (816). It may be desirable to keep the VMM loaded in memory to enable it to quickly virtualize the hardware again later. However, if the memory used for the system was borrowed from an OS, the VMM may also be unloaded by returning the memory to that OS for its use. This last step will likely be carried out using a kernel module or device driver within the OS, as described above.